目录

[开发文档 1](#_Toc27945)

[项目背景 1](#_Toc21816)

[需求分析 1](#_Toc8357)

[程序设计 2](#_Toc16444)

[总结 9](#_Toc30159)

[附录 10](#_Toc2626)

# 开发文档

## 项目背景

本项目旨在模拟实现一个结合C++标准库和Linux系统调用的Shell程序。

## 需求分析

功能需求

Shell程序能够提供命令的输入，执行并显示执行结果的功能。

Shell程序能够提供可shell编程的功能，能够执行简单的shell脚本。

Shell程序能够提供I/O重定向和管道的功能。

非功能需求

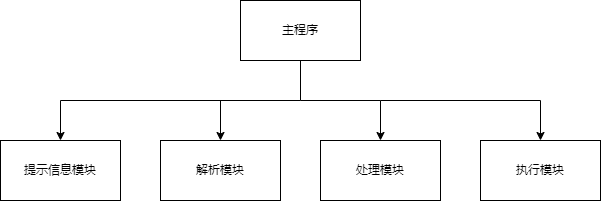
性能：能在合理的时间内执行用户输入的命令，必要时提供反馈信息。

可修改性：能在不较大地改变程序结构的情况下扩展程序功能。

功能性：系统能完成期望的工作

## 程序设计

组织结构



基于模块化设计思想，程序分为四个模块：提示信息模块、解析模块、处理模块、执行模块。

提示信息模块主要提供Shell程序需要展示的必要信息。

解析模块对用于输入的命令进行分割、将它们分割为单个命令。

处理模块主要处理命令中的重定向和管道等特殊命令。

执行模块将处理好的命令根据情况分别执行。

实现细节

main.cpp

//执行命令行字符串

void execute\_line(const std::string & str)

prompt.cpp

//获取系统信息 "主机名@用户名:当前目录$ "

std::string get\_prompt\_info()

parse.cpp

//解析命令行字符串，将其分割成单个命令参数

std::vector<std::string> parse\_command(const std::string & str)

//解析包含管道的命令行字符串，将其分割成多个命令

std::vector<std::vector<std::string>> parse\_pipe(const std::string & str)

handle.cpp

//处理重定向

void handle\_redirection(std::vector<std::string> & arg\_vec)

//处理管道

void handle\_pipe(std::vector<std::vector<std::string>> & command\_vec)

command.cpp

//cd

void builtin\_cd(const std::vector<std::string> & arg\_vec)

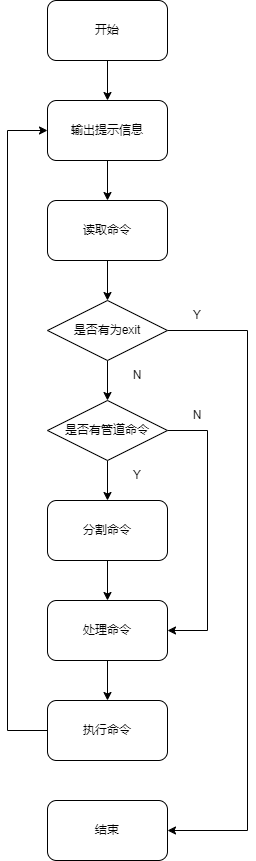
//外部命令

void external\_command(const std::vector<std::string> & arg\_vec)

//运行命令

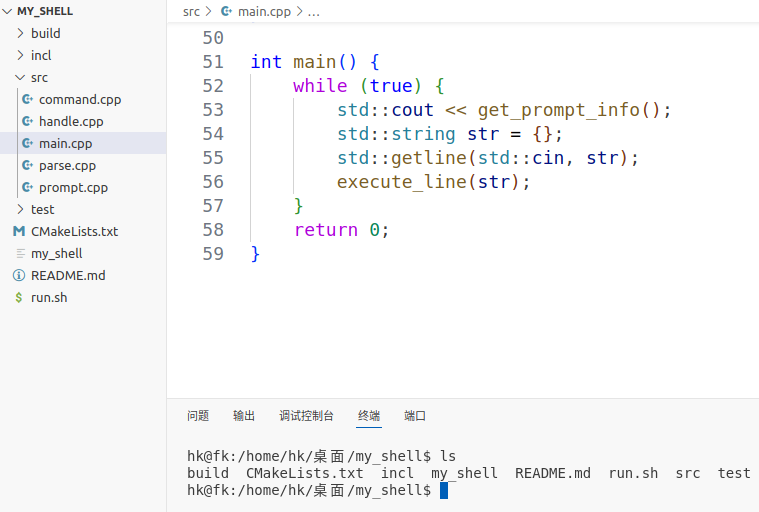
void run\_command(const std::vector<std::string> & arg\_vec)

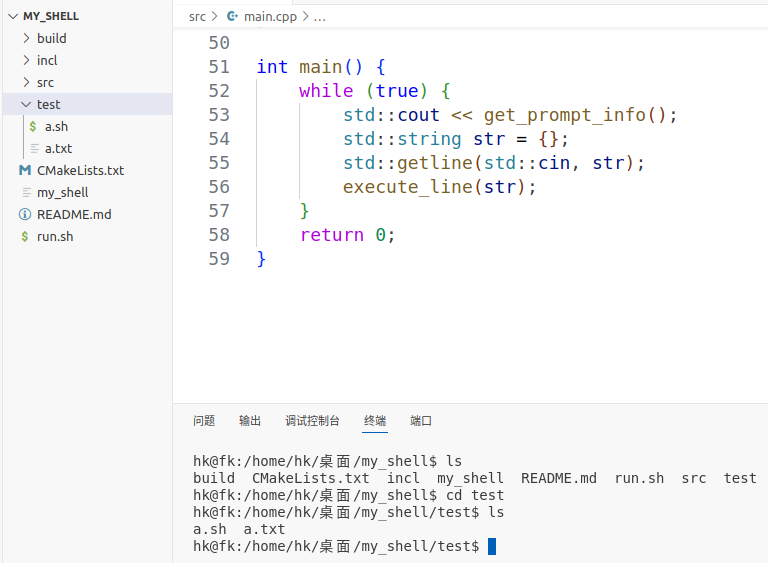
流程图



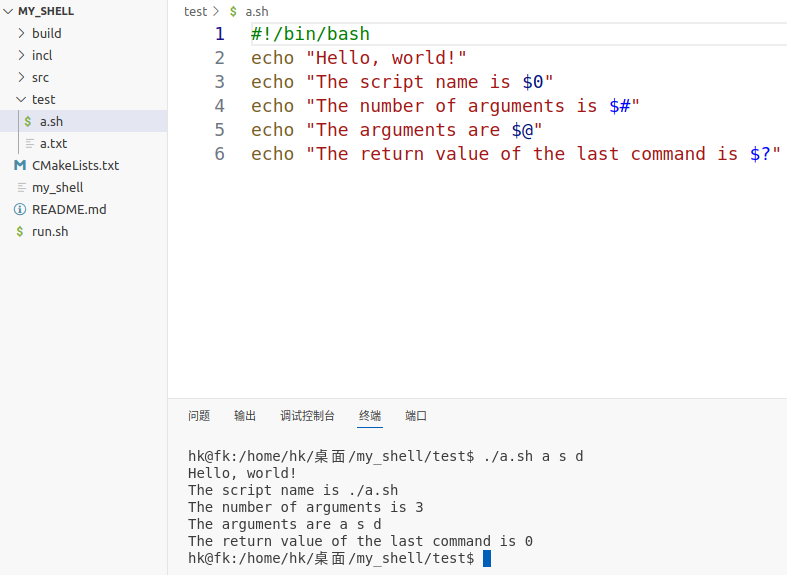
运行截图

基本命令

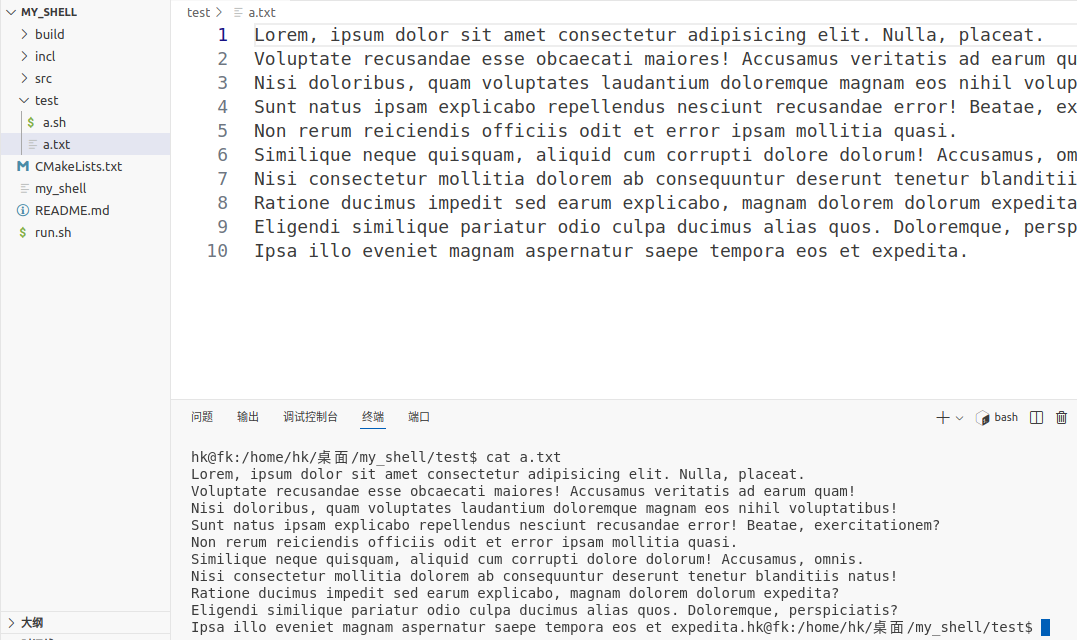




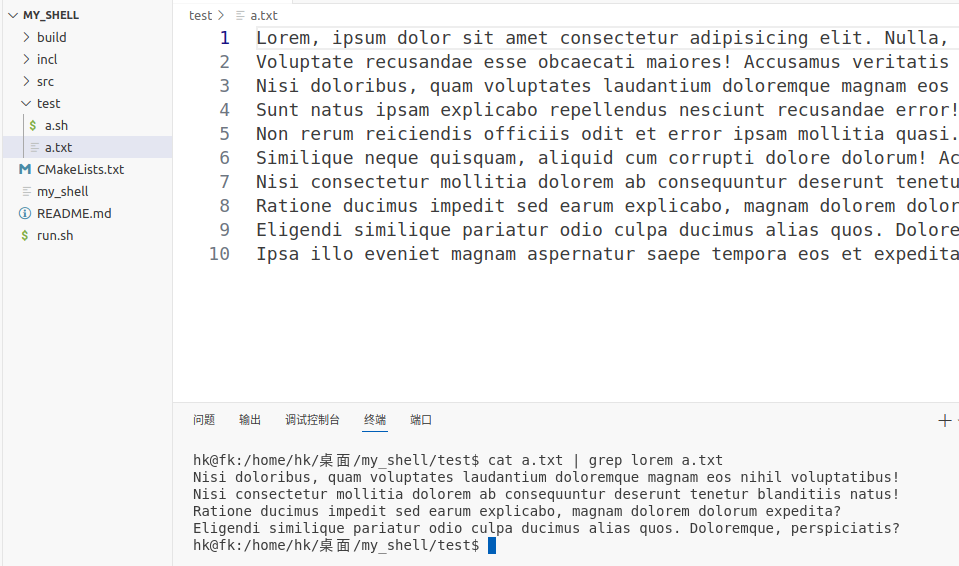
编写并运行Shell脚本

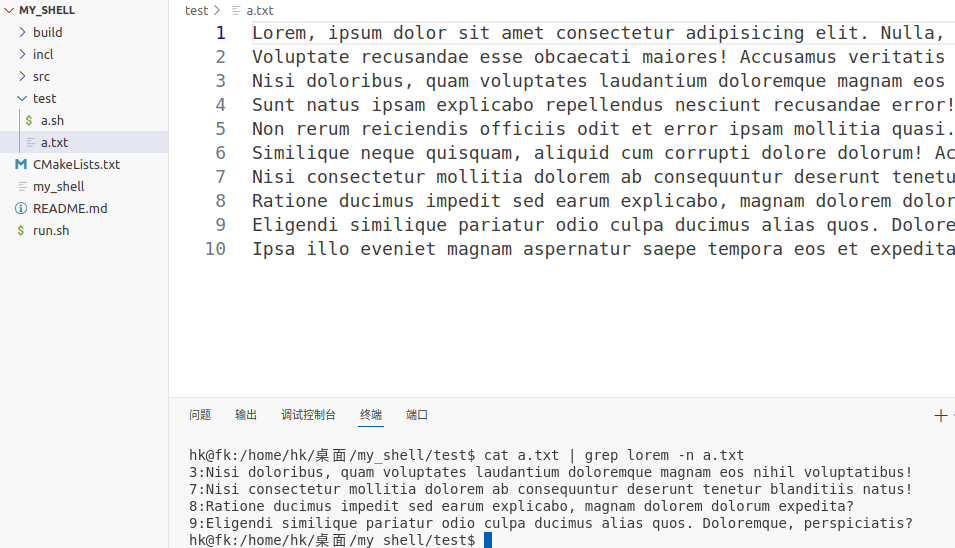


显示文件内容



管道命令

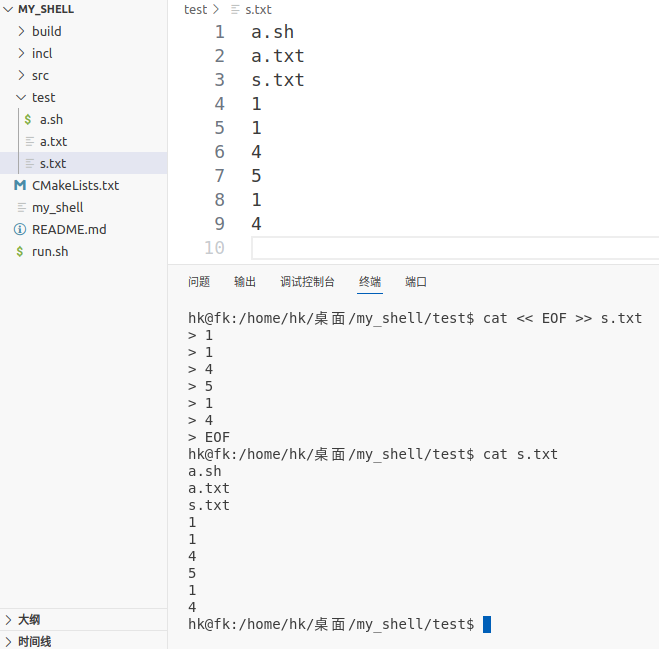




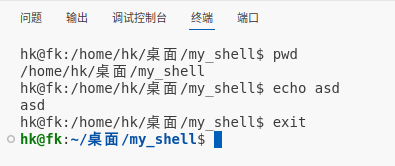
输出重定向



Here Document 重定向 + 追加重定向



其他命令



## 总结

学会使用CMake对项目进行构建和管理，提高程序开发效率。

学会使用C++标准库提供的新特性和Linux的内置函数编写程序。

加强对Shell常用命令的熟悉，深入理解Shell程序执行流程。

## 附录

参考书籍：《Unix-Linux 编程实践教程》

参考网站：C++开发手册：https://zh.cppreference.com/w/%E9%A6%96%E9%A1%B5